Online Learning From Incomplete and Imbalanced Data Streams

Dianlong You, *Member, IEEE*, Jiawei Xiao, Yang Wang, Huigui Yan, Di Wu, *Member, IEEE*, Zhen Chen, Limin Shen, *Member, IEEE*, and Xindong Wu, *Fellow, IEEE*

***Abstract*—Learning with streaming data has attracted extensive research interest in recent years. Existing online learning approaches have specific assumptions regarding data streams, such as requiring fixed or varying feature spaces with explicit patterns and balanced class distributions. While the data streams generated in many real scenarios commonly have arbitrarily incomplete feature spaces and dynamic imbalanced class distributions, making existing approaches be unsuitable for real applications. To address this issue, this paper proposes a novel Online Learning from Incomplete and Imbalanced Data Streams (OLI ^{2} DS) algorithm. OLI ^{2} DS has a two-fold main idea: 1) it follows the empirical risk minimization principle to identify the most informative features of incomplete feature spaces, and 2) it develops a dynamic cost strategy to handle imbalanced class distributions in real-time by transforming F-measure optimization into a weighted surrogate loss minimization. To evaluate OLI ^{2} DS, we compare it with state-of-the-art related algorithms in three kinds of experiments. First, we adopt 14 real datasets to simulate three scenarios of incomplete feature spaces, i.e., trapezoidal, feature evolvable, and capricious data streams. Second, based on a benchmark online analyzer, we generate 13 datasets to simulate incomplete data streams with different imbalance ratios. Third, we analyze concept drift in two simulated scenes, i.e., online learning and data stream mining, and verify the adaption of OLI ^{2} DS on repeated concept drifts and variable imbalance ratios. The results demonstrate that OLI ^{2} DS achieves a significantly better performance than its rivals. Besides, a real-world case study on movie review classification is conducted to elaborate on our OLI ^{2} DS algorithm's effectiveness. Code is released at https://github.com/youdianlong/OLI2DS.**

***Index Terms*—Data streams, F-measure, incomplete feature spaces, imbalanced data, online learning.**

I. Introduction

ONLINE learning [1], [2], [3] is an efficient and effective approach to handling high-dimensional data streams. Traditional online learning methods assume that data streams have a fixed, stable, and predefined feature space[4]. While in many real-world applications such as ecological monitoring systems[5], real-time network intrusion detection [6], disease screening and diagnosis[7], and spam e-mail classification[8], such assumption does not hold because their feature spaces usually keep varying with the continuous growth of instances. Hence, online learning from doubly-streaming data (i.e., streaming features and instances) is a new paradigm of data stream analytics that has thrived very recently.

In general, doubly-streaming data has three important inherent characteristics. The first characteristic is the arbitrarily incomplete feature space, where pre-existing features may become unobservable or vanish as the instance increases. For example, the features describing the patients' symptoms are usually collected from different inspection equipments (pulse monitors, thermometers, respiratory sensors, etc.) and healthcare service providers (labs, insurance companies, hospitals, etc.)[7]. The collected features are commonly incomplete with arbitrary missing values due to various reasons, e.g., different medical properties of individuals, relief or aggravation of symptoms, equipment failures, uncontrollable factors, etc. The second characteristic is the dynamic imbalanced class distribution. For example, in COVID-19 screening and diagnosis[9], only a few people are placed in the sick class and the majority are normal class. Meanwhile, with the pandemic spread or control, such imbalanced class distribution dynamically changes over time. The third characteristic is that concept drift often occurs, such as sudden and gradual drifts due to arbitrary changes in data streams[10], [11]. Therefore, the problem of online learning from doubly-streaming data is how to characterize the arbitrarily incomplete feature space, the dynamic imbalanced class distribution, and the adaptation to concept drift simultaneously.

Recently, a few related studies have attempted to address such problems, including Online Learning with Streaming Features (OLSF) [12], Feature Evolvable Streaming Learning (FESL) [13], Online Learning from Varying Features (OLVF) [14], and Generative Learning with Streaming Capricious(GLSC) [15], Online Bagging and Boosting for Imbalanced Data Streams (OBBIDS)[16], and Cost-Sensitive Adaptive Random Forest (CSARF)[17]. However, they only find a partial solution: 1) OLSF and FESL can only process the particular trapezoidal and

Manuscript received 7 December 2021; revised 19 December 2022; accepted 23 February 2023. Date of publication 3 March 2023; date of current version 15 September 2023. This work was partially supported in part by the National Natural Science Foundation of China under Grants 62276226, 62102348, 62176070, and 62120106008, in part by Hebei Natural Science Foundation under Grants F2021203038, and F2022203012, and in part by State Scholarship Fund of China Scholarship Council under Grant 202108130089. Recommended for acceptance by L. Nie. *(Corresponding author: Di Wu.)*

Dianlong You, Jiawei Xiao, Yang Wang, Huigui Yan, Zhen Chen, and Limin Shen are with the School of Information Science and Engineering, Yanshan University, Qinhuangdao, Hebei 066004, China (e-mail: youdianlong@sina.com; xiaojiaweix@163.com; wangyang_ysu@163.com; rocky_yhg@163.com; zhenchen@ysu.edu.cn; shenllmm@sina.com).

Di Wu is with the College of Computer and Information Science, Southwest University, Chongqing 400715, China (e-mail: wudi.cigit@gmail.com).

Xindong Wu is with the Key Laboratory of Knowledge Engineering with Big Data (the Ministry of Education of China), Hefei University of Technology, Hefei, Anhui 230009, China (e-mail: xwu@hfut.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TKDE.2023.3250472, provided by the authors.

Digital Object Identifier 10.1109/TKDE.2023.3250472

evolvable feature patterns, respectively; 2) OLVF and GLSC are designed for varying feature spaces with a balanced class distribution; 3) OBBIDS and CSARF are imbalanced online learning algorithms but only suitable for complete feature spaces. To the best of our knowledge, no existing online learning algorithms can fully characterize the arbitrarily incomplete feature space and the dynamic imbalanced class distribution. In the end, even though concept drift, one of the main characteristics of data streams, may potentially be addressed by online learning, there is no specific research on the adaptation to concept drift under doubly-streaming data and incomplete feature space.

Motivated by this situation, this paper explores a new online learning problem, termed Online Learning from Incomplete and Imbalanced Data Streams (OLI ^{2} DS). The challenges of implementing OLI ^{2} DS lie in two aspects: 1) how to employ the existing features of incomplete feature spaces to learn a classifier, and 2) how to design a dynamic optimization mechanism to adaptively match the dynamic class distribution and concept drift. To this end, we propose a novel OLI ^{2} DS algorithm with a three-fold main idea: 1) it follows the empirical risk minimization principle to identify the most informative features of incomplete feature spaces, and 2) it develops a dynamic cost strategy to handle imbalanced class distributions in real-time by transforming F-measure optimization into a weighted surrogate loss minimization, and 3) it uses the mechanisms of updating learner in real-time, feature space confidence, and dynamic misclassification cost to improve the adaptation to concept drift. Specifically, OLI ^{2} DS can react to drift instances rapidly by iteratively updating the learner; The measure of feature space confidence can reflect drifts according to the information changes of instances; The convergence of loss can be accelerated by using dynamic cost, which enables learners quickly adapt to drifts. The main contributions of this study are as follows.

1) This is the first study to explore a new online learning problem where doubly-streaming data have arbitrarily incomplete feature space, dynamic imbalanced class distribution, and frequent concept drift.
2) A novel Online Learning from Incomplete and Imbalanced Data Streams (OLI ^{2} DS) algorithm is proposed to address the raised new problem.
3) Theoretical analyses are provided for the proposed OLI ^{2} DS algorithm, including algorithm design, time complexity, upper bound of the cumulative hinge loss, and error rate bounds.
4) Extensive empirical studies on 14 real and 31 synthetic datasets are conducted to evidence our proposed algorithm's effectiveness, viability, and superiority.

II. Related Work

Online learning operates the learning paradigm of instances arriving one by one in a streaming fashion [4], [18]. Theoretically, since online learning updates the model in real-time, it can naturally fit concept drift to a certain extent. Unfortunately, under doubly-streaming data and incomplete feature space, the existing online methods lack a specific mechanism and demonstration for concept drift. This study explores a new online learning problem of OLI ^{2} DS. We relate the problem to two research thrusts, i.e., online learning from incomplete feature spaces and imbalance learning. We integrate the adaptive strategies of concept drift into the above two thrusts. This section presents related prior studies of each thrust and explains their limitations.

A. Online Learning From Incomplete Feature Spaces

This research thrust aims at characterizing the arbitrarily incomplete feature space to train a classifier that can make correct predictions for instances arriving in the form of streams. Current related algorithms include OLSF [12], FESL [13], OLVF [14], and GLSC [15]. OLSF learns from trapezoidal data streams wherein both instances and features keep increasing in a monotonic manner. However, trapezoidal data streams are the special patterns that are a little less common in practice[12]. FESL learns from the feature evolvable data streams wherein feature spaces evolve in batches. It minimizes a least square loss to learn a linear projection matrix from old feature spaces to new feature spaces [13]. However, such a projection does not carry any useful information. OLVF learns two classifiers to mine varying feature spaces. The instance classifier learns to classify the arriving instances, and the feature space classifier estimates the probability that the instance classifier makes a correct prediction. However, when the incompleteness of features is large, the feature space classifier may dramatically damage the performance of the instance classifier[14]. GLSC completes the unobserved features by constructing a generative graphical model, where a latent matrix is maintained to establish the correlations between observed features. However, the constructed graphical model may cause redundant completed features. Moreover, the computational cost of maintaining a latent matrix is extremely high[15].

In comparison, our OLI ^{2} DS algorithm does not make any assumptions regarding the patterns of incomplete features. It can characterize the most informative observed features by measuring the feature space confidence, making it not suffer from these limitations of the above-related algorithms. Besides, our OLI ^{2} DS algorithm has the strategies to handle the dynamic imbalanced class distribution that is not considered by the above-related algorithms.

B. Imbalance Learning

Imbalance learning [19] aims to overcome the effects of unequal misclassification costs and class imbalance in the learning process. Existing imbalance learning methods can be grouped into three categories, i.e., data-level, algorithm-level, and hybrid approaches. First, the sampling-based algorithm [20], [21] is an easy way to develop data-level approaches. Its principle is to alter the distribution of training data to decrease imbalance. Second, the principle of algorithm-level approaches is to modify the underlying learning or decision process to reduce the bias of the majority group by using the class penalty or weight strategy [22]. Cost-sensitive learning is a representative algorithm-level approach [23], and many related models have been proposed. For example, Wang et al. [16] constructed several online cost-sensitive bagging or boosting algorithms by modifying some ensemble-based imbalance learning algorithms to handle imbalanced data streams. Loezer et al. [17] designed a cost-sensitive CSARF to address the imbalanced data streams with complete feature spaces. Zhang et al. [24] proposed a cost-sensitive method using deep reinforcement learning to control the imbalanced costs of transactions and risk in financial portfolio tasks. Third, hybrid approach combines data-level and algorithm-level techniques in various ways, such as EasyEnsemble and BalanceCascade [25], SMOTEBoost [26], WHMBoost [27], and JOUS-Boost [28].

As discussed above, we see that most existing imbalanced learning algorithms are not designed for online learning. Although few studies of [16] and [17] can process imbalanced data streams, they require the feature change complete. In comparison, our OLI ^{2} DS algorithm aims at learning from imbalanced data streams with arbitrarily incomplete feature spaces that cannot be handled by [17] and [16].

C. Data Stream Mining

Data stream is a sequence of data instances rapidly arriving in a massive volume, which can be looked at only once and is unbound in size [10]. With the continuous arrival of instances, the distribution will change in real-time. These changes in the data can be considered, namely concept drift[11]. Real-time processing and concept drift are two inherent problems for streaming data. Data stream mining is to extract hidden knowledge from the continuous data streams. The ability to detect understand, and adapt to changes in the distribution of instances is paramount for data stream mining algorithms[10]. Additionally, one of the goals of data stream mining in classification is to efficiently deal with concept drift and skewed data [29]. In contrast, the goal of online learning is to make a sequence of accurate predictions given knowledge of the correct answer to previous prediction and possibly additional available information [30], [31].

To sum up, in online learning, the environment chooses instances in a full feedback or adversarial manner[30] while in data stream mining, the distribution changes with the input of instances.

III. Problem Settings

We focus on constructing an online learning model to address Incomplete and Imbalanced Data Streams (I ^{2} DS) by updating learner in real-time to improve performance and adapt to concept drift. We consider the binary classification problem on I ^{2} DS. The OLI ^{2} DS algorithm is designed to address I ^{2} DS by training a classifier. The adopted symbols and notations are summarized in Table S1 ”Symbols and descriptions” in the Supplementary File (available online).

A. Definition Settings

Definition 1 (Online Learning from Data Streams, OLDS). Let data stream $D = \{(x_t, y_t)|t \in \{1, 2, \ldots, T\}\}$ be a sequence of arrival training instances, where $x_t \in \mathbb{R}^{d_t}$ is a vector of d_t dimensions and $y_t \in \{-1, +1\}$ is the true label of x_t. We define *OLDS* as that, at each iteration, only a single instance x_t in D can be observed first, and then the y_t is revealed. Based on this, the learner is updated by measuring the loss between the prediction and the answer.

Definition 2 (Incomplete Data Streams, IDS). Given D, let $F_t = \{f_1, f_2, \ldots, f_{d_t}\}, F_t \in \mathbb{R}^{d_t}$ be the feature set that is carried by x_t. For any two instances x_i and x_j, $F_i = F_j$ does not always hold. In other words, the feature space can vary arbitrarily. Let $U_t = \{F_1 \cup F_2 \cup \cdots \cup F_t\}, U_t \in \mathbb{R}^{u_t}$ be the universal feature set till iteration t, which represents the set of all features that have been introduced, where $\mathbb{R}^{u_t}$ satisfies $u_t \leq d_1 + d_2 + \cdots + d_t$. Then the missing feature ratio φ of x_t can be computed by $1 - d_t/u_t$. We define that the D is an incomplete data stream.

Definition 3 (Incomplete and Imbalanced Data Streams, I ^{2} DS). Given D, if D is incomplete, and satisfies $n_- \gg n_+$ and the imbalance ratio (*IR*)$\leq 10\%$, we define that the D is an incomplete and imbalanced data stream. Note that we regard positive instances as the minority class and negative instances as the majority class in this article. Then *IR* is computed by (1), in which let n_- represent the number of majority class instances (i.e., $y_t = -1$) and n_+ represent the number of minority class instances (i.e., $y_t = +1$).

$$IR = \frac{n_+}{n_+ + n_-}. \tag{1}$$

B. Online Learning from I ^{2} DS

Without loss of generality, we consider that the online learning model for addressing I ^{2} DS can be framed in the objective as

$$\min_{\Phi_1,\ldots,\Phi_T} \frac{1}{T}\sum_{t=1}^{T} \ell(y_t; \Phi(z_t)) + \Omega(\Phi_t), \tag{2}$$

$$s.t. \prod(x_t; \mathbb{R}^c, \mathbb{R}^n) \overset{\text{i.i.d}}{\sim} z_t, \tag{3}$$

We generalize the objective in a sequence as follows: a) The constraint (3) assumes that the data sequence x_1, ..., x_T is independently inscribed from the unknown distribution, and we employ linear projections for a latent representation of the original data. b) As shown in (2), an online learner is trained on the latent representations z_t. To cope with the curse of dimensionality, a regularization term $\Omega(\Phi_t)$ is applied to the learning process to encourage a sparse solution. The details of which are given in Section IV.

We defined the form of the classifier as shown in (4), in which w_t is the weight vector of the classifier. z_t is the latent representations of x_t about $\mathbb{R}^c$ and $\mathbb{R}^n$.

$$\Phi(z_t) = w_t \cdot z_t. \tag{4}$$

For a I ^{2} DS, the above mentioned D is its sequence of arrival training instances. The specific task and steps for OLI ^{2} DS are as follows: 1) Our task is to construct the update rules of the classifier w_t, a weight vector at the tth iteration, on the current instance x_t, $w_t \in \mathbb{R}^{u_t}$. Noted that only current (x_t, y_t) can be observed at each iteration; 2) We construct the loss function ℓ_t at tth iteration based on the hinge loss to train the w_t; 3) After the prediction, we suffer an instantaneous loss between the prediction $\hat{y}_t$ and the y_t, and the w_t are aggressively updated depending on the loss ℓ_t; Meanwhile, 4) we use the informativeness of different parts as feature space weights and measure the imbalance of classes through the class imbalance rate (IR), which in turn focuses more on the learning of minority classes; 5) At the end of tth iteration, we update the w_t based on the loss and the empirical risk minimization principle.

C. Challenges of OLI ^{2} DS

Challenge I. Difficulty of learning from data streams with an incomplete feature space. In incomplete data streams, the feature space changes in unpredictable ways, where the feature set can introduce new or old features, and some existing features can vanish. Such a highly dynamic environment leads to the prediction performance deterioration of the learner. Since the feature set of any two instances may be completely different, it is difficult for the learner to make predictions about the instances by making full use of the previously learned knowledge.

Challenge II. Difficulty of learning from data streams with an imbalanced class distribution. In imbalanced data streams, the number of instances in one class overwhelms the number of instances in the other class. However, the overall data distribution is unknown because training instances arrive sequentially over time and only the old data can be observed by the learner. Since the learner cannot define the appropriate learning parameters in advance, this setting makes the learning process difficult. In most cases, the minority class appears more important than the majority class for a learning algorithm.

IV. OLI ^{2} DS Algorithm

A. Learning From Incomplete Feature Spaces

In incomplete data streams, the feature spaces of any two consecutive instances may differ, which badly lowers the classification performance. This is because no prior knowledge can be provided by the new feature space while only the known feature space can be used in the prediction. Furthermore, the vanished features cannot provide any useful information for updating the classifier. To sum up, online learning from incomplete feature spaces is difficult.

To make an accurate prediction, the feature space of x_t will be divided into three parts—vanished, common, and new feature space x_t^v, x_t^c, x_t^n—depending on whether the feature under consideration existed in the x_t or the previous instance. We denote w_t^v, w_t^c, w_t^n as the projection of the w_t on the x_t^v, x_t^c, x_t^n at the tth iteration, respectively.

Most importantly, an instance with more informativeness is more advantageous for optimizing learning model[32], [33]. In view of this, we propose an adaptive weighted strategy based on uncertainty to learn from dynamic incomplete feature spaces. A fact is that if the variance of a feature is large over iterations, it indicates that the feature has high uncertainty. Meanwhile, the feature can provide more information to optimize the model. Additionally, the variance is only related to the instance and is not affected by other external factors. Thus, we substitute uncertainty with variance to estimate the informativeness of a feature. In particular, after projecting a training instance onto a feature space, we calculate the cumulative mean of the informativeness of features as the confidence of the feature space, which can be formulated as the weight of the feature space in prediction results. At iteration t, denoted by h_t^i, the informativeness of the ith feature in instance x_t, as shown in Table ”S1 Symbols and descriptions” in Supplementary File, available online, and the confidence p_t^c of features on a common feature space is defined as

$$p_t^c = \sum_{i=1}^{d_t^c} \frac{h_t^i}{\sum\limits_{j=1}^{d_t} h_t^j}, \tag{5}$$

where d_t^c is the dimension of the common feature space. This notation and definition also apply to the confidence p_t^n of features on a new feature space. Then, features with more informativeness, i.e., more important features, will be updated with higher weights.

When an instance is received, a classifier is re-weighted on the basis of the confidence of features on the different feature spaces and generates a prediction. We modify the hinge loss to train a classifier w_t and define the loss ℓ_t of the classifier at iteration t as

$$\ell_t = \ell(y_t; \hat{y}_t) = \max\{0, 1 - y_t\,(p_t^c \cdot w_t^c \cdot x_t^c + p_t^n \cdot w_t^n \cdot x_t^n)\}. \tag{6}$$

To minimize the cumulative loss in online learning tasks, the weights of an instance are constructed and adapted incrementally after observing each instance, which may be so sensitive to noise that it causes overfitting and poor generalization since the classifier needs to predict correctly for each instance. To address this limitation, a soft-margin technique [34] has been widely used, which introduces a slack variable ξ, $\xi \in [0, 1)$. The idea of (7) is that by introducing ξ to remain nonlinearity, which tolerates the classifier a few mistakes so as to improve the generalization. The optimization problem is then defined as

$$w_{t+1} = \mathop{\arg\min}_{w:\ell_t \leq \xi} \frac{1}{2}\|w - w_t\|^2 + C\xi, \tag{7}$$

where $C > 0$ adjusts the slack variable ξ. Thus, using (6) and (7), our learning task can be formulated as a constrained optimization problem:

$$\begin{aligned} w_{t+1} = &\mathop{\arg\min}_{\substack{w=[w^v,w^c,w^n]:\\ \ell_t \leq \xi, \xi \geq 0}} \frac{1}{2}\|w^v - w_t^v\|^2 \\ &+ \frac{1}{2}\|w^c - w_t^c\|^2 + \frac{1}{2}\|w^n\|^2 + C\xi. \end{aligned} \tag{8}$$

Equation (8) tends to retain as much knowledge as possible learned from arrived instances by minimizing the difference between the weights of the current classifier and the classifier at the previous moment. The mechanism of iterative updating can also ensure that w_t can react to drift instances rapidly. Using the Lagrangian function with K.K.T. conditions [35] to solve (8), along with the inequality constraints defined in (8), we obtain

$$L(w,\xi,\tau,\eta) = \frac{1}{2}\|w^v - w_t^v\|^2 + \frac{1}{2}\|w^c - w_t^c\|^2 + \frac{1}{2}\|w^n\|^2 + C\xi + \tau(\ell_t - \xi) - \eta\xi, \tag{9}$$

where τ and η are Lagrange multipliers. To find the solution of $L(w,\xi,\tau,\eta)$, we differentiate $L(w,\xi,\tau,\eta)$ with respect to w_t^v, w_t^c, w_t^n, and ξ, yielding the following conditions:

$$\begin{aligned} w^v &= w_t^v \\ w^c &= w_t^c + \tau p_t^c y_t x_t^c \\ w^n &= \tau p_t^n y_t x_t^n \\ \eta &= C - \tau, \end{aligned} \tag{10}$$

Substituting (10) into (9), we obtain

$$L(\tau) = \frac{1}{2}\tau^2 (p_t^c)^2 \|x_t^c\|^2 + \frac{1}{2}\tau^2 (p_t^n)^2 \|x_t^n\|^2 + \tau(1 - y_t(p_t^c \cdot w_t^c \cdot x_t^c + p_t^n \cdot w_t^n \cdot x_t^n)), \tag{11}$$

$$\tau = \min\left\{C, \frac{\ell_t}{(p_t^c)^2 \|x_t^c\|^2 + (p_t^n)^2 \|x_t^n\|^2}\right\}. \tag{12}$$

After obtaining the solution to the optimization problem in (8), the update rule is derived as follows:

$$\begin{aligned} w_{t+1} &= [w_{t+1}^v, w_{t+1}^c, w_{t+1}^n] \\ &= \left[w_t^v, w_t^c + \min\left\{C p_t^c y_t x_t^c, \frac{\ell_t p_t^c y_t x_t^c}{(p_t^c)^2 \|x_t^c\|^2 + (p_t^n)^2 \|x_t^n\|^2}\right\},\right. \\ &\quad \left.\min\left\{C p_t^n y_t x_t^n, \frac{\ell_t p_t^n y_t x_t^n}{(p_t^c)^2 \|x_t^c\|^2 + (p_t^n)^2 \|x_t^n\|^2}\right\}\right]. \end{aligned} \tag{13}$$

B. Learning From Incomplete and Imbalanced Data Streams

The basic form of online learning is designed to optimize classification accuracy. The goal is to minimize cumulative loss and the number of misclassified instances on datasets. Classifiers will prefer the majority class if they neglect the imbalanced problem of classes. In the presence of imbalance classes, a common approach is to maximize F-measure. We wish to construct an online classifier that optimizes F-measure as well as performs structural [36], [37] and empirical risk minimizations [38], [39]. To that end, we observe that F_β-measure can be expressed as

$$F_\beta = \frac{(1+\beta^2)\, TP}{\beta^2 (TP + FN) + TP + FP}, \tag{14}$$

where TP, FN denote the number of instances predicted as positive and negative, respectively, when the actual class is positive. TN, FP denote the number of instances predicted as negative and positive, respectively, when the actual class is negative. P is the number of positive instances that have appeared. We replace TP with $P - FN$ and derive the F_β-measure maximization problem as follows:

$$\begin{aligned} \max F_\beta &\Rightarrow \min\left[1 + \frac{\beta^2 FN + FP}{(1+\beta^2)(P - FN)}\right] \\ &\Rightarrow \min[\beta^2 FN + FP - (1+\beta^2)(P - FN)] \\ &\Rightarrow \min[(1+2\beta^2)FN + FP - (1+\beta^2)P]. \end{aligned} \tag{15}$$

where $(1+\beta^2)P$ is a constant. From (15), we can know that maximizing F_β-measure is equivalent to

$$\min[(1+2\beta^2)FN + FP]. \tag{16}$$

FN shows more importance than FP. However, in online learning tasks, the distribution of majority and minority classes may change over time since training instances arrive one by one. Moreover, a learner wishes to reduce the number of misclassified instances for both classes. Thus, we transform (16) into a weighted form and define it as

$$\min(c_{10} FN + c_{01} FP), \tag{17}$$

where c_{10} and c_{01} denote the importance (weight) of FN and FP, respectively. However, optimizing FN and FP directly is often difficult as they are non-convex. Thus we achieve FN and FP optimization by minimizing the surrogate function—the hinge loss. Consequently, (17) can be expressed as

$$\begin{aligned} &\min\left[c_{10}\sum_{t=1}^{P} I(y_t \neq sign(w_t \cdot x_t) | y_t = +1)\right. \\ &\qquad \left. + c_{01}\sum_{t=1}^{N} I(y_t \neq sign(w_t \cdot x_t) | y_t = -1)\right] \\ &= \min\left[c_{10}\sum_{t=1}^{P} \ell_t(x_t, w_t; y_t = +1)\right. \\ &\qquad \left. + c_{01}\sum_{t=1}^{N} \ell_t(x_t, w_t; y_t = -1)\right] \\ &= \min c_t \sum_{t=1}^{T} \ell_t(x_t, w_t; y_t), \end{aligned} \tag{18}$$

where $I(\cdot)$ is an indicator function. Because only a single instance can be observed at iteration t, we replace c_{10} or c_{01} used by c_t, which denotes the weight of the class that instance belongs to iteration t. And then, we consider the F_β-measure maximization problem into the constrained optimization problem. Therefore, by combining (8) and (18), we obtain

$$w_{t+1} = \mathop{\arg\min}_{\substack{w=[w^v,w^c,w^n]:\\ c_t\ell_t \leq \xi, \xi \geq 0}} \frac{1}{2}\|w^v - w_t^v\|^2 + \frac{1}{2}\|w^c - w_t^c\|^2 + \frac{1}{2}\|w^n\|^2 + C\xi. \tag{19}$$

Note that c_t scales the classification loss, and the classifier is updated based on the loss, thus, c_t can be viewed as the cost of the class of the instance at iteration t. Typically, the cost of a class is correlated with the class distribution of data streams. To dynamically adjust the class cost as data flow in, a dynamic strategy is developed to adapt the importance of the class in real-time and effectively learn from imbalanced data streams. For concreteness, the dynamic strategy adaptively transforms the cost of each class with the distribution of known labels. Let $\phi(y_t)$ be an activation function defined as

$$\phi\left(y_t\right) = \begin{cases} 1, & \text{if } y_t = +1 \\ 0, & \text{if } y_t = -1 \end{cases}. \tag{20}$$

Then, we implement the dynamic cost c_t and define it as

$$c_t = \frac{1}{\left(\frac{n_+}{n_-}\right)^{\phi(y_t)} + \left(\frac{n_-}{n_+}\right)^{(1-\phi(y_t))}}. \tag{21}$$

As $c_t \in (0, 1)$, merely applying c_t in learning algorithms can lead to slow convergence. To avoid this, we make the learning step controllable by introducing a scaling factor θ. Then, we redefine the dynamic cost c_t and rewrite it as

$$c_t = \frac{\theta}{\left(\frac{n_+}{n_-}\right)^{\phi(y_t)} + \left(\frac{n_-}{n_+}\right)^{(1-\phi(y_t))}}, \tag{22}$$

where n_+, n_- represent the number of minority and majority instances that were observed, respectively, which can be obtained through tracking in real-time. Similar to (8), we solve the optimization problem in (19) and obtain

$$\begin{aligned} w^v &= w_t^v \\ w^c &= w_t^c + \tau c_t p_t^c y_t x_t^c \\ w^n &= \tau c_t p_t^n y_t x_t^n \\ \eta &= C - \tau \\ \tau &= \min\left\{ C, \frac{\ell_t}{c_t\left(\left(p_t^c\right)^2 \|x_t^c\|^2 + \left(p_t^n\right)^2 \|x_t^n\|^2\right)} \right\}. \end{aligned} \tag{23}$$

We can obtain the following update rules:

$$\begin{aligned} w_{t+1} &= [w_{t+1}^v, w_{t+1}^c, w_{t+1}^n] \\ &= \left[w_t^v, w_t^c + \min\left\{ C c_t p_t^c y_t x_t^c, \frac{\ell_t p_t^c y_t x_t^c}{\left(p_t^c\right)^2 \|x_t^c\|^2 + \left(p_t^n\right)^2 \|x_t^n\|^2} \right\}, \right. \\ &\left. \min\left\{ C c_t p_t^n y_t x_t^n, \frac{\ell_t p_t^n y_t x_t^n}{\left(p_t^c\right)^2 \|x_t^c\|^2 + \left(p_t^n\right)^2 \|x_t^n\|^2} \right\} \right], \end{aligned} \tag{24}$$

where c_t dynamically adjusts the cost for each class with only parameter θ, which reduces the time of cost search and difficulty of parameter search. When concept drift occurs, the cumulative mean of the uncertainty will change significantly. As the weight of (24), p_t^c and p_t^n can reflect the change in (5) and improve the adaptability of OLI ^{2} DS to concept drift. Meanwhile, with c_t is dynamic adjusted by n_+, n_-, and θ control step, the ℓ_t can make w_t follow the concept drift.

C. Model Sparsity

Since the dimension of data streams is enormous, all features are preserved in a classifier, which can lead to poor performance. Such as the classifier can make a false prediction because it cannot adjust the mutation of data in time when the relevance between a feature and a class label compared to the past is disparate. On the other hand, since all features used to make predictions may incur high computational and memory costs, we retain relatively important features by updating and truncating w_t in real-time. However, merely truncating the smallest weight in the classifier is crude. Another widely used choice is truncating the classifier after projecting it onto its L_1 ball. L_1 ball indicates the norm ball of the 1-norm of the classifier vector. Note that the truncation strategy described above can have a prejudice against features that are appeared in a few instances in incomplete data streams because they have small weights and are truncated easily. Furthermore, minor changes in the weight of a feature with a high degree of uncertainty can produce a different outcome. It is, therefore, important to retain the most important features while avoiding prejudice against features that are appeared in a few instances.

To accomplish this, we introduce relative uncertainty into the feature selection process. When the feature shows more uncertainty, the weight should be retained as much as possible. The strategies of dynamic updating, truncating features, and uncertainty can maintain classifier performance, which is more adaptable to distribution changes and concept drift. The following projection step is then introduced as

$$w_t = \min\left\{1, \frac{\lambda}{\langle w_t \cdot H_t \rangle}\right\} w_t, \tag{25}$$

where $H_t = [h_t^1, h_t^2, h_t^3, \ldots, h_t^{u_t}] \in \mathbb{R}^{u_t}$ denotes the relative uncertainty vector of the universal feature space at the tth iteration, which is composed of the informativeness of all the features that have been observed. $\lambda > 0$ is a regularization parameter. After the projection step, we retain the largest elements based on a truncation parameter B, and $B \in (0, 1]$. Hence, in the next prediction, only the retained weights are used in the model, and a sparse is introduced.

D. OLI^2 DS for Handling Concept Drift

To concept drift, OLI^2 DS ensures to adapt the unforeseeable changes in the underlying distribution of I^2 DS over time from three aspects.

1) Update w_t in real-time and dynamic select features for classifier by line 12. From lines 1-13 in Algorithm 1 and (24), OLI^2 DS iteratively updates the classifier w_t as the instances arrive one by one. This ensures that w_t can react to drift instances rapidly to a certain extent.

2) Use p_t^c and p_t^n to respond to drifts. Once concept drift occurs, the variances and uncertainty of the feature space of the instance will change significantly. As the cumulative mean of the uncertainty, p_t^c and p_t^n in (5) can reflect the change of concept drift. Therefore, in (24), p_t^c and p_t^n are used as the weight of the feature space in prediction, which can improve the adaptability of OLI^2 DS to concept drift.

3) Employ c_t to rapidly adapt drifts. From (22), (24) and Algorithm 1, the dynamic misclassification cost c_t can be dynamically adjusted between majority and minority class by dynamic n_+, n_- and θ for controlling learning step, then accelerate the convergence of ℓ_t. This can reduce the error ratio of w_t and

Algorithm 1. OLI^2 DS Algorithm

```
Input:
$C > 0$: Tradeoff parameter of OLI$^2$ DS
$\lambda > 0$: Regulariztion parameter of classifier
$B \in (0, 1]$: Proportion of selected features
$\theta > 0$: Scaling factor of dynamic cost
Initialize: $w_1 = (0, 0, \ldots, 0) \in \mathbb{R}^{d_1}$
for $t = 1, 2, \ldots, T$ do
    Receive instance $x_t \in \mathbb{R}^{d_t}$
    Identify the common and new feature spaces
        $\mathbb{R}^c = \mathbb{R}^{x_{t-1}} \cap \mathbb{R}^{x_t}, \mathbb{R}^n = \mathbb{R}^{x_t} - \mathbb{R}^c$
    Project $w_t, x_t$ onto $\mathbb{R}^c, \mathbb{R}^n$, respectively:
        $w_t^c, w_t^n, x_t^c, x_t^n$
    Calculate the confidence of features on the different
        feature spaces $p_t^c, p_t^n$ using (5)
    Predict the class label
        $\hat{y}_t = sign(p_t^c \cdot w_t^c \cdot x_t^c + p_t^n \cdot w_t^n \cdot x_t^n)$
    Receive the true label $y_t \in \{+1, -1\}$
    Suffer loss $\ell_t$ using (6)
    Calculate $c_t$ using (20) and (22)
    Update classifier with $\ell_t, y_t, c_t, p_t^c, p_t^n, x_t^c, x_t^n, C$ using
        (24)
    Project $w_t$ using (25) with $\lambda$
    Truncate $w_t$ based on $B$
end for
```

quickly respond to the change of data distribution when facing concept drift.

Through above mechanism of optimizing learner, OLI^2 DS can ensure the adaptation of concept drift. In the real scene of mining I^2 DS, we can record the change range of F-measure in running of OLI^2 DS. Once the decline of F-measure exceeds a threshold, it means that drift occurs. Then, the learner needs to receive instances with labels to update.

E. Algorithm Design and Complexity Analysis

We design our OLI^2 DS algorithm based on the above analyses. Due to the page limits, the flowchart and the detailed analysis of our algorithm are given in Section ”S2 FLOWCHART OF OLI^2 DS” in the Supplementary File, available online.

Furthermore, the pseudocode of OLI^2 DS is presented as Algorithm 1. For a single iteration, the time complexity of OLI^2 DS is $O(|w_t| + |x_t|)$, and the runtime of the OLI^2 DS is linear. A more detailed complexity analysis of OLI^2 DS is provided in Section ”S3 COMPLEXITY ANALYSIS” in the Supplementary File, available online.

V. Theoretical Analysis

In this section, we present an analysis of online learning from I^2 DS. Specifically, we first discuss the upper bound of the cumulative hinge loss of OLI^2 DS in a perfect case, where a learner can correctly predict each instance, then generalize and derive it as linearly inseparable. Finally, we provide OLI^2 DS error rate bounds for each class. Our bounds guarantee that our algorithm shows a lower cumulative hinge loss than the best fixed prediction, which is chosen in hindsight for any sequence of instances. More theoretical proofs of this section are provided in Section “S4 DERIVATIONS AND PROOFS” of the Supplementary File, available online.

When a learner makes a false prediction on an instance on iteration t, we have $y_t(p_t^c \cdot w_t^c \cdot x_t^c + p_t^n \cdot w_t^n \cdot x_t^n) < 0$, and the loss function $\ell_t > 1$. Thus, the cumulative hinge loss $\sum_{t=1}^{T} \ell_t$ is an upper bound of the number of misclassified instances. We denote the loss of off-line predictor at iteration t by ℓ_t^*, and ℓ_t^* is defined as follows:

$$\ell_t^* = \ell\left(\mathrm{w}^{x_t}; (x_t, y_t)\right), \tag{26}$$

where $\mathrm{w} \in \mathbb{R}^{u_T}$ represents an arbitrary vector and w^{x_t} represents the projection of w on x_t. This notation also applies to w^{w_t}, $\mathrm{w}^{w_{t+1}}$, $\mathrm{w}^{w_t^c}$, $\mathrm{w}^{w_t^n}$, $\mathrm{w}^{x_t^c}$ and $\mathrm{w}^{x_t^n}$. Then, we have Lemma 1 as follows.

Lemma 1. Let $(x_1, y_1), \ldots, (x_T, y_T)$ be a sequence of training data, where $x_t \in \mathbb{R}^{d_t}$ and $y_t \in \{+1, -1\}$ for all t. Let the dynamic cost $c_t = \theta/[(\frac{n_+}{n_-})^{\phi(y_t)} + (\frac{n_-}{n_+})^{1-\phi(y_t)}]$ and learning rate $\tau_t = \min\{C, \frac{\ell_t}{c_t((p_t^c)^2\|x_t^c\|^2 + (p_t^n)^2\|x_t^n\|^2)}\}$, as given in (22) and (23), respectively. The following bound holds for any $\mathrm{w} \in \mathbb{R}^{u_T}$:

$$\sum_{t=1}^{T} \tau_t c_t \{2\ell_t - \frac{2}{\varepsilon}\ell_t^* - \frac{2(\varepsilon - 1)}{\varepsilon} \\ - \tau_t c_t[(p_t^c)^2 \|x_t^c\|^2 + (p_t^n)^2 \|x_t^n\|^2]\} \leq \|\mathrm{w}\|^2. \tag{27}$$

First, we prove a loss bound of OLI^2 DS in the perfect case. There exists a classifier $\mathrm{w} \in \mathbb{R}^{u_T}$ that can make a prediction correctly over the sequences, i.e., $y_t(\mathrm{w}^{x_t} \cdot x_t) > 0$. By scaling the classifier $\mathrm{w} \in \mathbb{R}^{u_T}$, we have $y_t(\mathrm{w}^{x_t} \cdot x_t) > 1$, that is, a classifier $\mathrm{w} \in \mathbb{R}^{u_T}$ that can achieve zero hinge loss for all instances over T iterations. Therefore, Theorem 1 is the bound of the cumulative hinge loss of OLI^2 DS.

Theorem 1. Let $(x_1, y_1), \ldots, (x_T, y_T)$ be a sequence of training instances, where $x_t \in \mathbb{R}^{d_t}$, $y_t \in \{+1, -1\}$, and $\|x_t\|^2 \leq \mathrm{N}^2$ for all t. Assume that there exists a classifier w such that $\ell_t^* = 0$ for all t. The cumulative hinge loss of OLI^2 DS over the sequence satisfies

$$\sum_{t=1}^{T} \ell_t \leq \sqrt{\|\mathrm{w}\|^2\mathrm{N}^2 + \sum_{t=1}^{T}\left(\frac{\varepsilon - 1}{\varepsilon}\right)^2} + \sum_{t=1}^{T}\left(\frac{\varepsilon - 1}{\varepsilon}\right). \tag{28}$$

Next, we generalize Theorem 1 and derive a linearly inseparable case, where any vector $\mathrm{w} \in \mathbb{R}^{d_T}$ cannot perfectly separate training data. We have the bound of the cumulative hinge loss of OLI^2 DS as Theorem 2.

Theorem 2. Let $(x_1, y_1), \ldots, (x_T, y_T)$ be a sequence of training instances, where $x_t \in \mathbb{R}^{d_t}$, $y_t \in \{+1, -1\}$, and $\|x_t\|^2 = 1$ for all t. The following bound holds for any vector $\mathrm{w} \in \mathbb{R}^{u_T}$, which is the cumulative hinge loss of OLI^2 DS over the sequence:

$$\sum_{t=1}^{T} \ell_t \leq \sqrt{\|\mathrm{w}\|^2 + \sum_{t=1}^{T}\left(\frac{\ell_t^* + 1}{\varepsilon} - 1\right)^2} + \sum_{t=1}^{T}\left(\frac{\ell_t^* + 1}{\varepsilon}\right) - T. \tag{29}$$

TABLE I
STATISTICS OF THE DATASETS IN EXPERIMENTS

RQ	Data Category	Datasets	Instances	Features	IRs(%)	Datasets	Instances	Features	IRs(%)
RQ1	real datasets	splice	3190	60	48.1	spectrometer	531	278	8.5
		kr-vs-kp	3196	36	47.8	scene	2407	294	7.4
		credit-a	690	15	44.5	libras	360	90	6.7
		spambase	4601	57	39.4	thyroid	3772	52	6.1
		wdbc	569	30	37.3	dermatology	358	35	5.6
		svmguide3	1243	22	26.3	arrhythmia	452	93	5.5
		spect	267	22	20.6	car	1728	21	3.7
	online simulated data stream*	Data Stream	Instances	Features	Change	Stream Setting			

RQ2		class imbalance	200,000	100	static
					dynamic
						safe[N],borderline[N],rare/outlier[N],for N∈{5,15,30}
						sudden[N],gradual[N],for N∈{10→50,10→90}			

		Data Stream	Instances	Features	Drift Type	Drift Range	N(%)	Unlabeled Range	Generator
RQ3-1		online learning	20,000	100	sudden[N] gradual[N]	[10000,10001] [10000,12000]	{10,50}		RandomRBF, Imbalanced
		data stream mining	20,000	100	sudden[N] gradual[N]	[10000,10001] [10000,12000]	{10,50}	[11000,15000], [16000,18000]	RandomRBF, Imbalanced

* 13/4/14 datasets are generated for RQ2/RQ3-1/RQ3-2, respectively. Due to the page limits, the datasets of RQ3-2 are set in Table S2 of the Supplementary File.

The following theorem provides an error rate bound of OLI^2 DS on different classes.

Theorem 3. Let $(x_1, y_1), \ldots, (x_T, y_T)$ be a sequence of training instances, where $x_t \in \mathbb{R}^{d_t}$, $y_t \in \{+1, -1\}$, and $\|x_t\| \leq \mathrm{N}$ for all t. The following bound holds for any vector $\mathrm{w} \in \mathbb{R}^{u_T}$, which is the number of misclassifications of OLI^2 DS on any class over the sequence:

$$E_m \leq \max\left\{\frac{1}{C}, \mathrm{N}^2\right\}\left[\frac{\|\mathrm{w}\|^2}{c_t} + \frac{2C}{\varepsilon}\sum_{t=1}^{T}(\ell_t^* + \varepsilon + 1)\right]. \tag{30}$$

C is the tradeoff parameter in OLI^2 DS algorithm. c_t represents the cost of the current class at iteration t, which will increase to update with more steps when the ratio of the current class is smaller in all instances that have been learned but will decrease to prevent the classifier from being dominated by the majority class when the ratio of the current class is greater.

VI. Experiments

The experiments in this section aim to answer the following research questions(RQs).

- RQ1. Does OLI^2 DS outperform other state-of-the-art incomplete feature space learning algorithms when learning from trapezoidal, feature evolvable, and capricious data streams?
- RQ2. How does OLI^2 DS perform compared to other state-of-the-art online imbalance learning algorithms when learning from imbalanced data streams with incomplete feature space?
- RQ3. 1) Can OLI^2 DS effectively handle concept drift on online learning scene and data stream mining scene? 2) Can OLI^2 DS effectively adapt I^2 DS with repeated concept drifts and variable IRs?

A. General Settings

In this section, we simply introduce the general settings. The detailed settings are shown in Section "S6.1 Supplementary General Settings" of the Supplementary File, available online.

Datasets. To answer RQ1, we simulate three different scenes using 14 UCI[1] datasets: data with simulated trapezoidal data streams, feature evolvable streams, and capricious data streams, which are used in the benchmark algorithms. These datasets had been normalized before the experiments. For RQ2 and RQ3, we produce 13 imbalanced data streams with different types of data difficulties and 18 data streams with concept drift using MOA[40] generators. The MOA add-in[2] was used, which allows users to simulate imbalanced data streams with different distributions and imbalance ratios [41]. Each data stream combines two types of difficult factories. Table I summarizes the statistics of datasets and data streams.

Evaluation Metrics. In this paper, we evaluate the performance of algorithms using F_β-measure (abbr., F-measure, where β=1 as default) and G-mean. Additionally, the results of the Kappa, Matthews Correlation Coefficient (MCC), Recall, Precision, and Cumulative Error Rate (CER) are provided in the Supplementary File, available online.

Baselines. We compare OLI^2 DS with 8 related state-of-the-art online learning algorithms. Specifically, we compare OLI^2 DS with OLSF, FESL, OLVF, and GLSC in trapezoidal data streams, feature evolvable streams, and capricious data streams, respectively. To better evaluate the performance of OLI^2 DS in high-dimensional incomplete and imbalanced data streams, we use four online imbalance learning algorithms as baselines. Table S3 gives brief descriptions of all compared methods in the Supplementary File, available online.

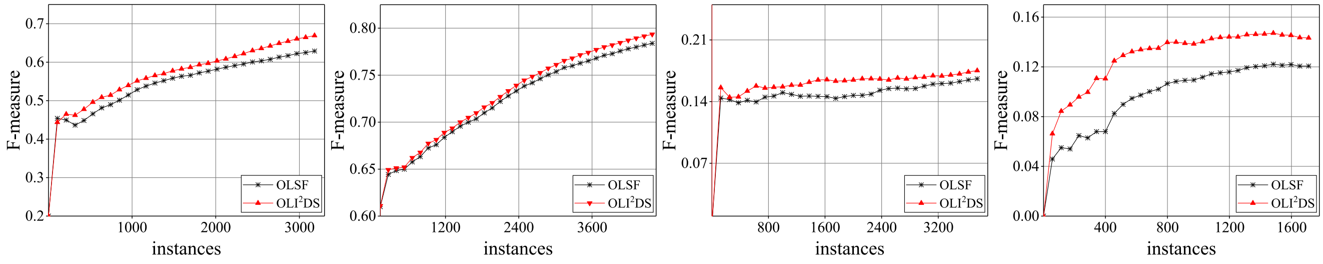
Implementation Details. To ensure fairness, we simulate different data scenes including, trapezoidal data streams, feature evolvable streams, and capricious data streams by following the settings described in [12], [13], [15], respectively. In these scenes, we apply a random permutation and repeat it 10 times to obtain the average, which is also used in OLSF, FESL, GLSC, and OLVF. When discussing OLI^2 DS compared with online imbalance learning algorithms, we use the implementation in MOA[40], where each algorithm uses the Naive Bayes as the base learner, and ten base learners per ensemble, as recommended by their authors. In Table S3 of the Supplementary File, available online, the OAC2, OCSB2, ORUSB, and OUOB are

[1] Datasets can be found in http://archive.ics.uci.edu/ml/
[2] https://github.com/dabrze/imbalanced-stream-generator

TABLE II
EXPERIMENTAL RESULTS (MEAN F-MEASURE ± STANDARD DEVIATION, MEAN G-MEAN ± STANDARD DEVIATION) ON 14 DATASETS IN SIMULATED TRAPEZOIDAL DATA STREAMS AND FEATURE EVOLVABLE STREAMS

	Trapezoidal Data Streams				Feature Evolvable Streams			
	F-Measure (↑)		G-Mean (↑)		F-Measure (↑)		G-Mean (↑)	
Dataset	OLSF	OLI^{2}DS	OLSF	OLI^{2}DS	FESL	OLI^{2}DS	FESL	OLI^{2}DS
splice	.687±.005●	**.705±.007**	.680±.004●	**.701±.006**	.740±.001●	**.765±.000**	0.727±0.001●	**.763±.002**
kr-vs-kp	.630±.010●	**.670±.009**	.659±.006●	**.691±.007**	.795±.002●	**.878±.001**	.795±.002●	**.878±.001**
credit-a	.618±.018●	**.643±.011**	.642±.017●	**.666±.012**	.512±.023●	**.572±.015**	.587±.016●	**.626±.013**
spambase	.785±.004●	**.793±.005**	.819±.004●	**.827±.004**	.565±.001●	**.638±.002**	.110±.007●	**.695±.002**
wdbc	.909±.007●	**.915±.009**	.931±.005●	**.936±.008**	.666±.018●	**.910±.007**	.731±.016●	**.919±.005**
svmguide3	.499±.017●	**.514±.015**	.651±.014●	**.668±.013**	.454±.027●	**.531±.011**	.587±.020●	**.683±.010**
spect	.673±.018●	**.699±.021**	.658±.026●	**.687±.019**	.693±.014●	**.722±.009**	.687±.012●	**.706±.012**
spectrometer	.259±.022●	**.283±.016**	.686±.034●	**.723±.022**	.320±.001●	**.374±.019**	.777±.003●	**.821±.019**
scene	.184±.009●	**.195±.003**	.604±.017●	**.626±.006**	.011±.004●	**.229±.000**	.072±.024●	**.689±.000**
libras	.171±.020●	**.191±.013**	.610±.042●	**.651±.024**	.154±.018●	**.252±.015**	.294±.020●	**.755±.024**
thyroid	.169±.006●	**.176±.003**	.631±.012●	**.646±.007**	.008±.003●	**.203±.002**	.062±.021●	**.690±.004**
dermatology	.247±.022●	**.494±.054**	.787±.031●	**.894±.033**	.044±.133●	**.300±.009**	.057±.172●	**.850±.008**
arrhythmia	.132±.009●	**.185±.009**	.564±.024●	**.697±.019**	.143±.010●	**.166±.003**	.571±.012●	**.657±.007**
car	.122±.012●	**.144±.003**	.667±.036●	**.729±.006**	.124±.022●	**.149±.002**	.347±.034●	**.741±.004**
Avg.	.435±.013	**.472±.013**	.685±.019	**.724±.013**	.373±.020	**.478±.007**	.457±.026	**.748±.008**
w/t/l	14/0/0	—	14/0/0	—	14/0/0	—	14/0/0	—

● indicates OLI^{2}DS outperforms its rivals statistically significantly (hypothesis supported by paired t-tests at 0.05 significance level), and ○ indicates a tie. The win/tie/loss counts for OLI^{2}DS are summarized in the last row, abbreviated as w/t/l



(a) kr-vs-kp (b) spambase (c) thyroid (d) car

Fig. 1. Trends of the average F-measure of OLSF and OLI^2 DS algorithms on trapezoidal data streams on 4 datasets. Due to the page limits, all 14 datasets are shown in Fig. S2 of the Supplementary File, available online.

the abbreviations of OnlineAdaC2, OnlineCSB2, OnlineRUSBoost, and OnlineUnderOverBagging, respectively. Each algorithm is evaluated in a prequential test with a sliding window size of 1,000 instances. For each dataset, when comparing any two algorithms, we use the paired t-tests to show the difference statistically over ten runs with a significance level of 0.05.

B. Comparisons Analysis on Incomplete Data Streams (RQ1)

We analyze the experimental results of F-measure, G-mean, Runtime, Kappa, MCC, Recall, and Precision on three kinds of data streams. Additionally, the runtime analyses of algorithms are placed in Section "S6.2 Supplementary Experimental Results on Incomplete Data Streams" of the Supplementary File, available online. Furthermore, we perform the analysis of CER and the ablation study for parametric analysis of our method, which is provided in Section "S7 COMPARISONS ANALYSIS OF CER" and "S8 ABLATION STUDY" of the Supplementary File, available online, specifically.

1) Experiments on Trapezoidal Data Streams: Table II compares the experimental results of OLI^2 DS and OLSF algorithms on trapezoidal data streams. Based on these results, we can deduce that OLI^2 DS outperforms the OLSF on all datasets when the same settings are used. Note that in trapezoidal data streams, the vanished and common feature spaces never change, and only the confidence of the current training instance can be used in the prediction.

Moreover, OLI^2 DS achieves better performance as the level of class imbalance increases, as shown in Fig. 1 and Fig. S2 of the Supplementary File, available online. This is because OLI^2 DS considers the skewed class distribution in the optimization, whereas OLSF focuses only on minimizing the cumulative loss. Therefore, OLI^2 DS is more robust in classification performance.

2) Experiments on Feature Evolvable Streams: Table II shows the experimental results of the average F-measure and G-mean achieved using FESL[13] and our algorithm on feature evolvable streams. On feature evolvable streams, OLI^2 DS and FESL achieve 47.8% and 37% F-measure on average, respectively, and OLI^2 DS outperforms FESL on all datasets. This is because FESL learns a fixed mapping matric primarily through the period when old and new features coexist. However, the matrix may lead to incorrect feature reconstruction when the feature relationship changes or includes noise data.

From Fig. 2 and Fig. S3 of the Supplementary File, available online, OLI^2 DS outperforms FESL throughout the iteration. Furthermore, the performance of OLI^2 DS shows more stability than FESL as features evolve. The main reason is that OLI^2 DS

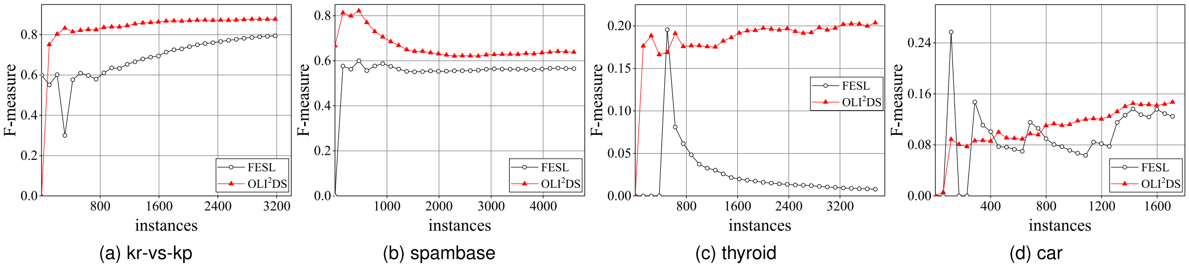


Fig. 2. Trends of the average F-measure of FESL and OLI^2 DS algorithms in simulated feature evolvable streams on 4 datasets. Due to the page limits, all 14 datasets are shown in Fig. S3 of the Supplementary File, available online.

TABLE III
EXPERIMENTAL RESULTS (MEAN F-MEASURE ± STANDARD DEVIATION, MEAN G-MEAN ± STANDARD DEVIATION) ON 14 DATASETS IN SIMULATED CAPRICIOUS DATA STREAMS

	F-Measure (↑)			G-Mean (↑)		
Dataset	GLSC	OLVF	OLI^{2}DS	GLSC	OLVF	OLI^{2}DS
splice	.752±.006●	.748±.007●	**.760±.006**	.748±.007●	.743±.008●	**.759±.006**
kr-vs-kp	**.836±.007**○	.793±.007●	**.836±.007**	**.842±.007**○	.800±.008●	**.842±.007**
credit-a	.783±.012●	.794±.011●	**.804±.011**	.798±.011●	.808±.009●	**.816±.012**
spambase	.851±.004●	.810±.003●	**.852±.004**	.877±.003●	.842±.003●	**.878±.003**
wdbc	.905±.010●	.909±.013●	**.929±.009**	.927±.008●	.930±.010●	**.947±.007**
svmguide3	.484±.007●	.495±.012●	**.522±.010**	.639±.007●	.650±.011●	**.676±.009**
spect	.674±.015●	.694±.018●	**.709±.020**	.674±.017●	.689±.015●	**.705±.019**
spectrometer	.234±.008●	.246±.009●	**.325±.012**	.607±.009●	.613±.006●	**.768±.013**
scene	.205±.006●	.146±.010●	**.220±.005**	.647±.011●	.536±.021●	**.673±.009**
libras	.185±.021●	.189±.011●	**.208±.013**	.637±.041●	.647±.021●	**.678±.026**
thyroid	.197±.008●	.076±.005●	**.210±.005**	.688±.017●	.408±.013●	**.714±.009**
dermatology	.256±.015●	.205±.025●	**.458±.059**	.807±.018●	.732±.043●	**.909±.027**
arrhythmia	.107±.016●	.080±.011●	**.152±.012**	.511±.041●	.443±.032●	**.626±.028**
car	.125±.006●	.125±.007●	**.141±.004**	.677±.018●	.677±.019●	**.724±.010**
Avg.	.471±.010	.451±.011	**.509±.012**	.720±.015	.680±.016	**.765±.013**
w/t/l	13/1/0	14/0/0	—	13/1/0	14/0/0	—

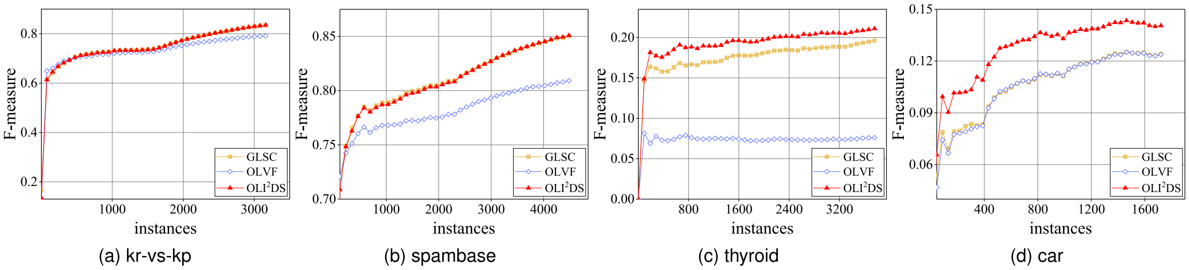


Fig. 3. Trends of the average F-Measure of GLSC, OLVF, and OLI^2 DS algorithms in simulated capricious data streams on 4 datasets. Due to the page limits, all 14 datasets are shown in Fig. S4 of the Supplementary File, available online.

explores and compares the importance of new and old feature spaces to modify the current classifier using the instances of an overlapping period, ensuring a smooth transition of classifier performance during feature evolution.

3) Experiments on Capricious Data Streams: The average performances of GLSC, OLVF, and OLI^2 DS algorithms on capricious data streams are reported in Table III. We observe that our OLI^2 DS outperforms both OLVF and GLSC algorithms on all imbalanced datasets and shows comparable, or even better, performance on balanced datasets. Note that GLSC trains classifiers with a graphical model, implying that GLSC requires far more instances and features to achieve comparable performance to the OLI^2 DS. Particularly, the results *kr-vs-kp* and *spambase*, support it.

As shown in Fig. 3 and Fig. S4 of the Supplementary File, available online, with increasing class imbalance, OLI^2 DS outperforms OLVF and GLSC. Moreover, OLI^2 DS achieves a better performance in a shorter time. This is because GLSC and OLVF only focus on dynamic feature spaces and minimizing total error while neglecting skewed class distributions.

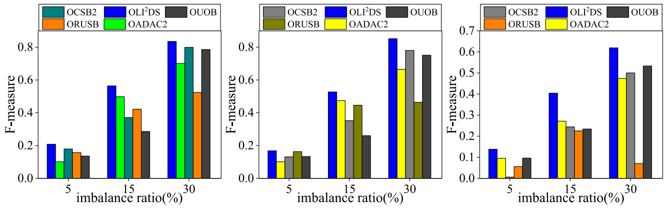
C. Comparisons Analysis on Class Imbalance (RQ2)

In this section, we study the performance of OLI^2 DS and four online imbalance algorithms on data streams with different static/dynamic imbalance ratios (*IR*s). We evaluate the performance using F-measure, G-mean, Kappa, MCC, Recall, and

TABLE IV
THE FINAL-STEP PERFORMANCE FROM OLI^2 DS AND FOUR ONLINE IMBALANCE ALGORITHMS ON DATA STREAMS WITH STATIC *IR*S

Data		F-measure (↑)					G-mean (↑)				
Distribution	IR	OLI^{2}DS	OAC2	OCSB2	ORUSB	OUOB	OLI^{2}DS	OAC2	OCSB2	ORUSB	OUOB
safe	30	**.835±.000**	.701±.018●	.799±.003●	.524±.002●	.786±.003●	**.915±.001**	.797±.001●	.850±.002●	.611±.001●	.856±.002●
	15	**.564±.002**	.498±.007●	.371±.004●	.422±.012●	.286±.031●	.849±.001	**.856±.007**○	.500±.004●	.615±.001●	.419±.003●
	5	**.208±.001**	.101±.004●	.179±.021●	.157±.017●	.136±.004●	**.764±.000**	.606±.032●	.321±.025●	.664±.002●	.279±.016●
borderline	30	**.852±.001**	.665±.011●	.780±.024●	.464±.002●	.751±.004●	**.926±.001**	.759±.010●	.836±.007●	.562±.020●	.825±.011●
	15	**.527±.000**	.474±.003●	.352±.003●	.446±.014●	.260±.022●	**.844±.000**	.833±.018●	.483±.032●	.619±.019●	.395±.040●
	5	**.168±.001**	.101±.003●	.131±.040●	.163±.019●	.133±.028●	**.759±.001**	.536±.006●	.268±.021●	.688±.019●	.278±.024●
rare/outlier	30	**.619±.001**	.474±.008●	.500±.008●	.070±.006●	.533±.031●	**.708±.001**	.404±.022●	.600±.012●	.191±.005●	.656±.003●
	15	**.404±.001**	.271±.006●	.244±.022●	.225±.014●	.234±.029●	**.685±.000**	.359±.004●	.402±.040●	.424±.015●	.372±.017●
	5	**.138±.001**	.095±.010●	.006±.002●	.056±.006●	.096±.004●	**.679±.001**	.060±.003●	.053±.011●	.332±.021●	.158±.009●
Avg. safe		**.536±.001**	.433±.010	.450±.009	.368±.010	.403±.013	**.843±.001**	.753±.013	.557±.010	.630±.001	.518±.007
Avg. borderline		**.516±.001**	.413±.006	.421±.023	.358±.012	.381±.018	**.843±0.001**	.709±.011	.529±.020	.623±.019	.499±.025
Avg. rare/outlier		**.387±.001**	.280±.008	.250±.011	.117±.009	.288±.021	**.691±.001**	.274±.010	.352±.021	.316±.014	.395±.010
w/t/l		—	9/0/0	9/0/0	9/0/0	9/0/0	—	8/1/0	9/0/0	9/0/0	9/0/0

● indicates OLI^{2}DS outperforms its rivals statistically significantly (hypothesis supported by paired t-tests at 0.05 significance level), and ○ indicates a tie. The win/tie/loss counts for OLI^{2}DS are summarized in the last row, abbreviated as w/t/l



(a) safe (b) borderline (c) rare/outlier

Fig. 4. The Final-Step F-measure from OLI^2 DS and four online imbalance algorithms on class imbalanced data stream with static *IR*s.

Precision. Details of the corresponding experimental results are reported below. Due to the page limits, the results of the Kappa, MCC, Recall, Precision, and Runtime are provided in Tables S5-S9 of the Supplementary File, available online.

1) Class Imbalance Analysis on Data Streams With Static Imbalance Ratio: Table IV shows their final-step F-measure and G-mean performance in given data streams with different difficult factors and imbalance ratios (*IR*s). We can see that OLI^2 DS outperforms other online imbalance algorithms under various *IR*s. The reason is that OLI^2 DS can adjust the misclassification cost of the current instance in a timely manner according to the current minority class ratio that has been observed, which greatly improves the minority class recall. Regardless of the distributions, the performance of all algorithms decreases as the data *IR* decreases. This suggests that complex minority class distributions and severely skewed class distributions will result in lower learner performance. It is obvious that the class with few instances (e.g., $IR = 5\%$) cannot provide sufficient information.

From Tables S5 and S6 in the Supplementary File, available online, we can see that OLI^2 DS exhibits outstanding performances for some evaluators, e.g., Kappa, MCC, Recall, and Precision under the different *IR*s. Experiments show that, by introducing soft-margin techniques, OLI^2 DS effectively avoids overly strict classification rules and results in better performance than other online imbalance stream learning algorithms, even at low *IR*s.

Fig. 4 compares the above-mentioned online imbalance methods for different minority class distributions under each *IR*. We can observe that as the *IR* decreases, all learners suffer severe performance degradation, and this phenomenon is further exacerbated when the data shows a rare/outlier distribution, particularly for the ensemble-based imbalance stream learning algorithm. This observation suggests that, in online learning tasks, severe data skew increases the learning difficulty of the model under rare/outlier distribution.

2) Class Imbalance Analysis on Data Streams With Dynamic Imbalance Ratio: Table V shows that OLI^2 DS significantly outperforms some benchmark algorithms for online imbalance learning on both the *IR* changes with sudden and gradual cases. Additionally, OLI^2 DS exhibits similar F-measure and G-mean both on the *IR* changes with sudden and gradual, as shown in Table V from average performance in the bottom. From Tables S7 and S8 in the Supplementary File, available online, we can see that OLI^2 DS remains outstanding performances in evaluators, e.g., Kappa, MCC, Recall, and Precision, with dynamic *IR*s. Moreover, the averages of sudden and gradual are similar for final-step results.

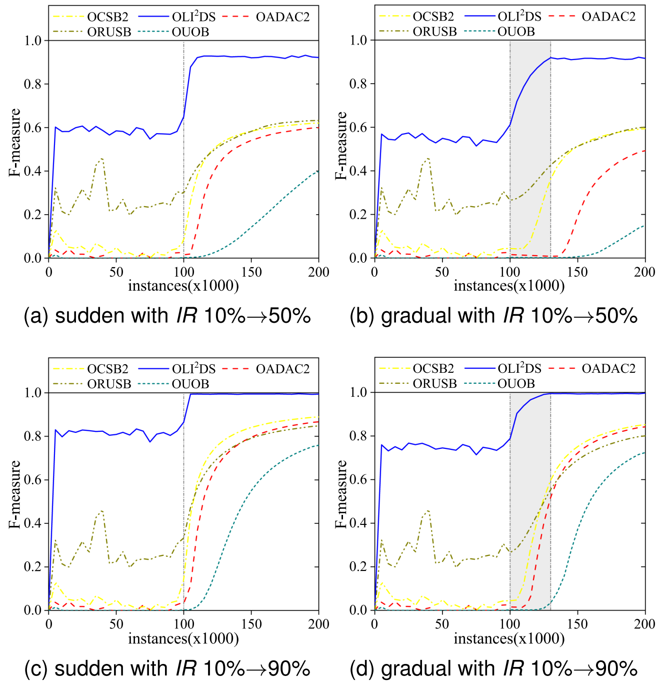
To analyze the impact of IRs in real-time, we record the prequential F-measure performances of OLI^2 DS and its rivals on data streams with dynamic *IR*, as shown in Fig. 5. For the sudden changes with *IR* 10%→90%, before the *IR* changes, OLI^2 DS has better F-measure performance than other methods, indicating that adaptively re-weight and dynamic cost strategy can handle the incomplete and imbalanced data streams. In Fig. 5(a) and (c), the curves of these methods increase steeply after the change happens. This is because the scale of the positive instances suddenly increases when the change happens. Similar trends can be seen in the gradual situation, as shown more clearly in the shaded areas of Fig. 5(b) and (d). These show that, compared with its rivals, OLI^2 DS can perceive changes more quickly and then quickly optimize classifiers. Moreover, this also explains why OLI^2 DS achieves similar performance in both sudden and gradual scenes.

D. Performance Analysis on Concept Drift of OLI^2 DS (RQ3)

We analyze our OLI^2 DS on I^2 DS regarding concept drift from two aspects: a) analyzing OLI^2 DS in both online learning and data stream mining scenes; b) analyzing OLI^2 DS on I^2 DS with repeated drifts and variable imbalance ratios (IRs).

TABLE V
THE FINAL-STEP PERFORMANCE FROM OLI^2 DS AND FOUR ONLINE IMBALANCE ALGORITHMS ON DATA STREAMS WITH DYNAMIC *IR*S

Data		F-measure (↑)					G-mean (↑)				
Distribution	IR	OLI^{2}DS	OAC2	OCSB2	ORUSB	OUOB	OLI^{2}DS	OAC2	OCSB2	ORUSB	OUOB
sudden	10→50	**.922±.001**	.569±.004●	.548±.002●	.614±.001●	.347±.010●	**.918±.001**	.682±.006●	.665±.002●	.726±.001●	.481±.009●
	10→90	**.994±.001**	.834±.006●	.867±.006●	.854±.010●	.780±.009●	**.967±.001**	.832±.005●	.871±.006●	.839±.009●	.797±.009●
gradual	10→50	**.916±.000**	.535±.002●	.345±.006●	.272±.009●	.147±.012●	**.907±.001**	.680±.001●	.473±.006●	.543±.0010●	.286±.014●
	10→90	**.997±.001**	.805±.003●	.811±.008●	.813±.010●	.695±.011●	**.973±.001**	.824±.001●	.831±.007●	.822±.012●	.734±.010●
Avg. sudden		**.958±.001**	.702±.005	.708±.004	.734±.006	.564±.010	**.943±.001**	.757±.006	.768±.004	.783±.005	.639±.009
Avg. gradual		**.957±.001**	.670±.003	.578±.007	.543±.010	.421±.012	**.940±.001**	.752±.001	.652±.007	.683±.011	.510±.012
w/t/l		—	4/0/0	4/0/0	4/0/0	4/0/0	—	4/0/0	4/0/0	4/0/0	4/0/0



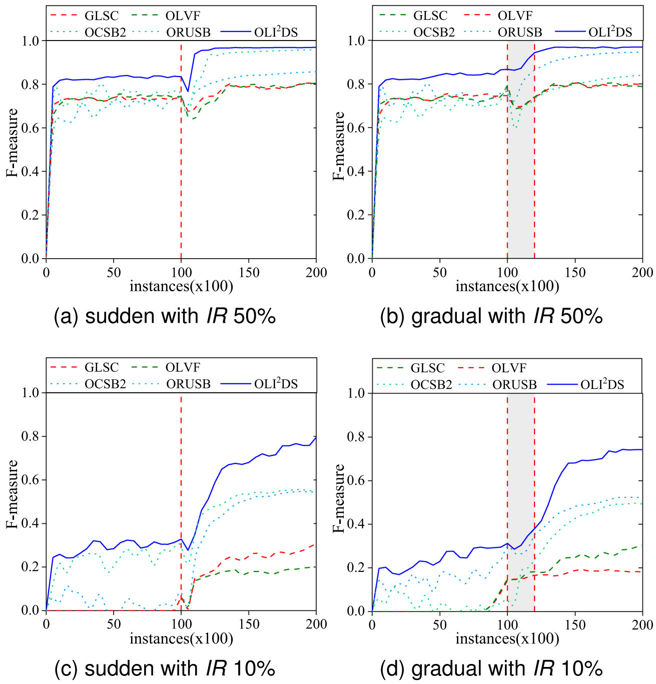
(a) sudden with *IR* 10%→50%

(b) gradual with *IR* 10%→50%

(c) sudden with *IR* 10%→90%

(d) gradual with *IR* 10%→90%

Fig. 5. The Final-Step F-measure from OLI^2 DS and four online imbalance algorithm on class imbalanced data stream with dynamic *IR*s.



(a) sudden with *IR* 50%

(b) gradual with *IR* 50%

(c) sudden with *IR* 10%

(d) gradual with *IR* 10%

Fig. 6. The performance on online learning scene for OLI^2 DS and its rivals from I^2 DS with concept drift and different *IRs*. The sudden drift in (a) and (c), gradual drift in (b) and (d); The dotted red line (x-axis: 100) denote the occurring location of sudden drift in (a) and (c); The range of gradual drift in (b) and (d): 100 → 120 on x-axis.

1) Performance Analysis on Online Learning and Data Stream Mining Scenes With Concept Drift (RQ3-1): Online learning takes place in a sequential way. On each round, a classifier receives an instance and makes a prediction. Then the classifier obtains the answer and calculates loss. In data stream mining, labeled and unlabeled instances alternately emerge. When unlabeled instances are received, the classifier is not updated and only predicts the results. The experiments are set with the sudden/gradual concept drift and IR= 10% or 50%, respectively. The synthesized datasets, as shown in Table I (RQ3-1), are generated by MOA with the attributes of drift type, drift location, drift range, unlabeled range, etc. Due to the page limits, other experimental results and analyses on the above two scenes, e.g., Kappa, MCC, Recall, and Precision are provided in Tables S10-S13 of the Supplementary File, available online.

Online Learning Scene. From Fig. 6, we note that: 1) OLI^2 DS is obviously superior to its rivals in all cases, due to our adaptive design for handling concept drift in OLI^2 DS; 2) Once drift occurs (x-axis: 100 in Fig. 6(a)–(d)), OLI^2 DS decreases slightly, and then, increases immediately. This indicates that OLI^2 DS is sensitive and adaptive to concept drift due to updating the classifier in real-time; 3) Sudden drift (Fig. 6(a) and (c)) has a severer impact on classifier performance than gradual drift (Fig. 6(b) and (d)). This is because when sudden drift occurs, the data distribution changes more rapidly than gradual drift.

From Table VI, we can observe that: 1) For sudden and gradual drifts, OLI^2 DS performs significantly better than its rivals. Especially, OLI^2 DS demonstrates high performance on F-measure and G-mean when IR=50%; 2) Under imbalanced data, i.e., IR=10%, and sudden/gradual drift, OLI^2 DS presents more significantly outstanding than its rivals. This also shows that OLI^2 DS is extremely effective in handling imbalanced data with concept drifts.

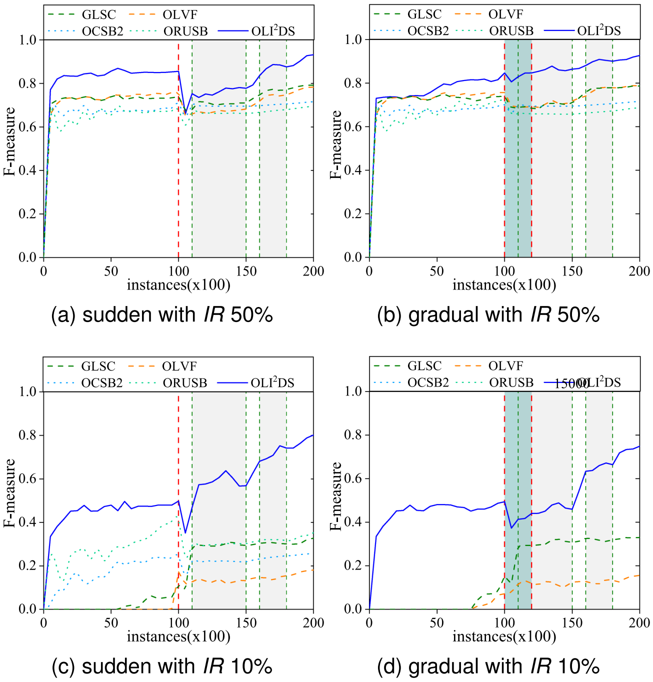
Data Stream Mining Scene. From Fig. 7, we note that: 1) OLI^2 DS is obviously superior to its rivals. Once drift occurs, OLI^2 DS decreases slightly, and then increases immediately, indicating significant adaptability; 2) When unlabeled instances (x-axis:

TABLE VI
THE FINAL-STEP F-MEASURE AND G-MEAN ON ONLINE LEARNING SCENE FOR OLI^2 DS AND ITS RIVALS FROM I^2 DS WITH CONCEPT DRIFT AND DIFFERENT *IRs*

Data		F-measure (↑)					G-mean (↑)				
Distribution	IR(%)	OLIDS	OLVF	GLSC	OCSB2	ORUSB	OLIDS	OLVF	GLSC	OCSB2	ORUSB
sudden drift	50	**.970**	.805	.810	.857	.958	**.971**	.707	.713	.671	.959
	10	**.796**	.200	.308	.545	.550	**.969**	.581	.649	.687	.826
gradual drift	50	**.970**	.798	.788	.840	.946	**.966**	.687	.697	.857	.947
	10	**.742**	.182	.304	.493	.520	**.955**	.556	.674	.647	.789
Avg. sudden drift		**.883**	.502	.559	.701	.754	**.970**	.644	.681	.679	.892
Avg. gradual drift		**.856**	.490	.546	.666	.733	**.961**	.622	.686	.752	.868

TABLE VII
THE FINAL-STEP F-MEASURE AND G-MEAN ON DATA STREAM MINING SCENE FOR OLI^2 DS AND ITS RIVALS FROM I^2 DS WITH CONCEPT DRIFT AND DIFFERENT *IRs*

Data		F-measure (↑)					G-mean (↑)				
Distribution	IR(%)	OLIDS	OLVF	GLSC	OCSB2	ORUSB	OLIDS	OLVF	GLSC	OCSB2	ORUSB
sudden drift	50	**.932**	.783	.792	.716	.695	**.929**	.692	.701	.671	.691
	10	**.802**	.181	.327	.259	.353	**.967**	.559	.708	.594	.647
gradual drift	50	**.927**	.789	.787	.716	.687	**.921**	.686	.673	.662	.686
	10	**.749**	.156	.330	.287	.357	**.953**	.510	.706	.438	.666
Avg. sudden drift		**.867**	.482	.560	.487	.524	**.948**	.626	.705	.632	.669
Avg. gradual drift		**.838**	.472	.558	.501	.522	**.937**	.598	.690	.550	.676



(a) sudden with *IR* 50%

(b) gradual with *IR* 50%

(c) sudden with *IR* 10%

(d) gradual with *IR* 10%

Fig. 7. The performance on data stream mining scene for OLI^2 DS and its rivals from I^2 DS with concept drift and different *IRs*. The sudden drift in (a) and (c), gradual drift in (b) and (d); The dotted red line (x-axis: 100) denote the occurring location of sudden drift; The range of gradual drift in (b) and (d): 100 → 120 on x-axis; The areas of in dotted green lines (x-axis: 110 → 150, and 160 → 180) denote instances are not labeled.

110→150 and 160→180 in Fig. 7(a)–(d)) are recieved, F-measure appears descending on the back end because the classifier only predicts and does not update in this area; 3) With new labeled instances arriving (x-axis: 150→160 and 180→200 in Fig. 7(a)–(d)), the F-measure of OLI^2 DS rises more immediately than its rivals. This means that OLI^2 DS can rapidly improve the performance of the classifier by using the adaptive weighted strategy; 4) Sudden drifts (drift location: 100 on x-axis in Fig. 7(a) and (c)) affect the classifier more rapidly than gradual drifts (x-axis: 100→120 in Fig. 7(b) and (d)). However, as instances continue to arrive, the classifier's performance will progressively advance.

From Table VII, we can see that: 1) OLI^2 DS achieves relatively stable and high-performance on F-measure and G-mean when IR=50%; 2) OLI^2 DS remains outstanding performances than its rivals in all cases, especially when IR=10%. It indicates that OLI^2 DS focuses on minority classes and realizes high performance under sudden and gradual drifts.

2) Performance Analysis on Repeated Concept Drifts and Variable IRs (RQ3-2): We present the experimental settings and results of OLI^2 DS on I^2 DS with repeated drifts and variable IRs in Section S11 of the Supplementary File, available online.

VII. A CASE STUDY ON MOVIE REVIEW EMOTIONAL TEXT CLASSIFICATION

In this section, we apply our algorithm to a real-world movie review scene to verify its effectiveness. Online multi-classification is difficult due to complex classification and unstable drift, etc. One solution is to convert multi-classification into binary classification in multi-labels. That is, if one class is considered positive and the others are considered negative, the binary classification will result in imbalanced class distribution. Specifically, we use the Internet Movie Database (IMDB[3]) constructed by Stanford Artificial Intelligence Laboratory. The dataset allows certain non-word tokens (e.g., "!" and ":-)") in vocabulary because they indicate sentiments. As shown in Fig. 8, movie reviews are generated continuously as data streams. Words in movie reviews can be regarded as features. Each column represents a review, which corresponds to a class label. The words with color boxes appear for the first time, and the number in front of each word indicates its importance. The

[3] https://www.imdb.com

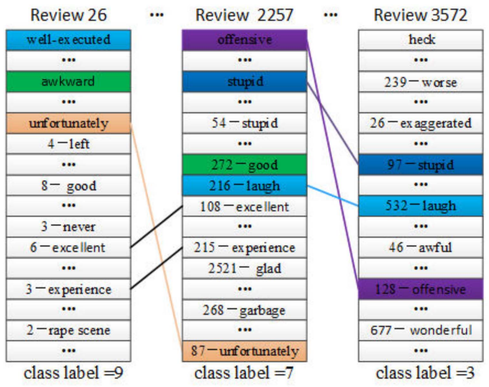


Fig. 8. A scenario of emotional texts in movie reviews.

TABLE VIII
PERFORMANCE COMPARISONS OF THREE ALGORITHMS ON MOVIE REVIEW EMOTIONAL TEXT CLASSIFICATION

Algorithm	F-measure (↑)	G-mean (↑)	Runtime(s) (↓)
GLSC	.156±.003	.500±.006	3359.87±30.234
OLVF	.159±.002	.508±.003	2123.43±8.958
OLI^{2}DS	**.260±.003**	**.662±.003**	**29.471±.372**

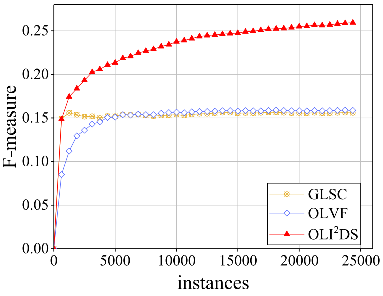


Fig. 9. Trends of average F-measure of OLI^2 DS, GLSC and OLVF algorithms in movie review dataset.

multi-classification labels ($\in \{1, 2, 3, \ldots, 10\}$) are mapped to binary classification label {+1, -1}.

The goal is to categorize each movie review into positive and negative sentiments using linearly mapping star values to $\{+1, -1\}$ as review labels. Additionally, we treat one of the star values (star $= 9$ in our experiment) as the positive sentiment and the other as the negative sentiment. Because the OLSF and FESL cannot handle varying feature spaces, we compare OLI^2 DS with GLSC and OLVF.

Table VIII reports the experimental results of three algorithms. We can find out that OLI^2 DS obtains the best average F-measure, G-mean, and runtime. This is because OLI^2 DS can learn the discrepancy between different feature spaces by exploring and measuring the uncertainty of observed features. Fig. 9 shows the trends of the F-measure of the three algorithms as data streams in. The OLI^2 DS achieves better performance as the amount of training data increases, whereas the other two algorithms stabilize at a lower value after a short convergence. The main reason is that GLSC reconstructs nonexistent features with the help of a feature graphical model, which tends to learn empirical knowledge and brings more errors as the number of features increases. Furthermore, the classifier is easily dominated by the majority instances because OLVF only focuses on optimizing the cumulative error.

VIII. CONCLUSION

This paper explores a new online learning problem where doubly-streaming data have arbitrarily incomplete feature space and dynamic imbalanced class distribution. To this end, we propose a novel Online Learning from Incomplete and Imbalanced Data Streams (OLI^2 DS) algorithm with the two-fold idea: 1) OLI^2 DS can identify the most informative features of arbitrarily incomplete feature spaces by following the empirical risk minimization principle, and 2) OLI^2 DS can handle imbalanced class distributions in real-time by transforming F-measure optimization into a weighted surrogate loss minimization to develop a dynamic cost strategy. In the experiments, we adopt 14 real datasets to simulate three scenes of incomplete feature spaces, i.e., trapezoidal, feature evolvable, and capricious data streams. Besides, we adopt a benchmark online analyzer to generate 13 datasets to simulate incomplete data streams with different imbalance ratios, and 18 datasets to simulate concept drifts. The results substantiate that our OLI^2 DS achieves a significantly better performance than the state-of-the-art related algorithms on the tested datasets. Although our OLI^2 DS can work in I^2 DS with concept drift, the concept drift still is a challenging and open problem in the above environments. In the future, we plan to systematically study the mechanisms of detection, understanding, and adaptation to more sophisticated concept drift under doubly-streaming data.

REFERENCES

[1] K. Bhatia and K. Sridharan, “Online learning with dynamics: A minimax perspective,” in *Proc. 34th Conf. Neural Inf. Process. Syst.*, 2020, pp. 15020–15030.

[2] W. Ma, “Projective quadratic regression for online learning,” in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 5093–5100.

[3] S. Mitra and A. Gopalan, “On adaptivity in information-constrained online learning,” in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 5199–5206.

[4] S. C. Hoi, J. Wang, and P. Zhao, “LIBOL: A library for online learning algorithms,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 495–499, 2014.

[5] X. Fu, E. Seo, J. Clarke, and R. A. Hutchinson, “Link prediction under imperfect detection: Collaborative filtering for ecological networks,” *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 8, pp. 3117–3128, Aug. 2021.

[6] A. Phadke, M. Kulkarni, P. Bhawalkar, and R. Bhattad, “A review of machine learning methodologies for network intrusion detection,” in *Proc. IEEE 3rd Int. Conf. Comput. Methodol. Commun.*, 2019, pp. 272–275.

[7] D. Zhang, M. Jin, and P. Cao, “ST-MetaDiagnosis: Meta learning with spatial transform for rare skin disease diagnosis,” in *Proc. IEEE Int. Conf. Bioinf. Biomed.*, 2020, pp. 2153–2160.

[8] S. Sel and D. Hanbay, “E-mail classification using natural language processing,” in *Proc. IEEE 27th Signal Process. Commun. Appl. Conf.*, 2019, pp. 1–4.

[9] M. Barstuan, U. Ozkaya, and S. Ozturk, “Coronavirus (COVID-19) classification using CT images by machine learning methods,” in *Proc. CEUR Workshop*, 2021, pp. 29–35.

[10] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, and G. Zhang, “Learning under concept drift: A review,” *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 12, pp. 2346–2363, Dec. 2019.

[11] S. Agrahari and A. K. Singh, “Concept drift detection in data stream mining: A literature review,” *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, pp. 9523–9540, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1319157821003062

[12] Q. Zhang, P. Zhang, G. Long, W. Ding, C. Zhang, and X. Wu, “Online learning from trapezoidal data streams,” *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 10, pp. 2709–2723, Oct. 2016.

[13] B.-J. Hou, L. Zhang, and Z.-H. Zhou, “Learning with feature evolvable streams,” *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 6, pp. 2602–2615, Jun. 2021.

[14] E. Beyazit, J. Alagurajah, and X. Wu, “Online learning from data streams with varying feature spaces,” in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 3232–3239.

[15] Y. He, B. Wu, D. Wu, E. Beyazit, S. Chen, and X. Wu, “Toward mining capricious data streams: A generative approach,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 3, pp. 1228–1240, Mar. 2021.

[16] B. Wang and J. Pineau, “Online bagging and boosting for imbalanced data streams,” *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 12, pp. 3353–3366, Dec. 2016.

[17] L. Loezer, F. Enembreck, J. P. Barddal, and A. de Souza Britto Jr, “Cost-sensitive learning for imbalanced data streams,” in *Proc. 35th Annu. ACM Symp. Appl. Comput.*, 2020, pp. 498–504.

[18] P. Zhao, D. Wang, P. Wu, and S. C. H. Hoi, “A unified framework for sparse online learning,” *ACM Trans. Knowl. Discov. Data*, vol. 14, no. 5, 2020, Art. no. 59.

[19] S. Wang, L. L. Minku, and X. Yao, “A systematic study of online class imbalance learning with concept drift,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 10, pp. 4802–4821, Oct. 2018.

[20] C.-L. Liu and P.-Y. Hsieh, “Model-based synthetic sampling for imbalanced data,” *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 8, pp. 1543–1556, Aug. 2020.

[21] L. Li, H. He, and J. Li, “Entropy-based sampling approaches for multi-class imbalanced problems,” *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 11, pp. 2159–2170, Nov. 2020.

[22] J. M. Johnson and T. M. Khoshgoftaar, “Survey on deep learning with class imbalance,” *J. Big Data*, vol. 6, no. 1, pp. 1–54, 2019.

[23] N. Thai-Nghe, Z. Gantne, and L. Schmidt-Thieme, “Cost-sensitive learning methods for imbalanced data,” in *Proc. IEEE Int. Joint Conf. Neural Netw.*, 2010, pp. 1–8.

[24] Y. Zhang, P. Zhao, Q. Wu, B. Li, J. Huang, and M. Tan, “Cost-sensitive portfolio selection via deep reinforcement learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 236–248, Jan. 2022.

[25] B. S. Raghuwanshi and S. Shukla, “Classifying imbalanced data using balancecascade-based kernelized extreme learning machine,” *Pattern Anal. Appl.*, vol. 23, no. 3, pp. 1157–1182, 2020.

[26] J. Zhang, T. Wang, W. W. Ng, and W. Pedrycz, “Ensembling perturbation-based oversamplers for imbalanced datasets,” *Neurocomputing*, vol. 479, pp. 1–11, 2022.

[27] J. Zhao, J. Jin, S. Chen, R. Zhang, B. Yu, and Q. Liu, “A weighted hybrid ensemble method for classifying imbalanced data,” *Knowl.-Based Syst.*, vol. 203, 2020, Art. no. 106087.

[28] D. Mease, A. J. Wyner, and A. Buja, “Boosted classification trees and class probability/quantile estimation,” *J. Mach. Learn. Res.*, vol. 8, no. 3, pp. 409–439, 2007.

[29] P. Wang, N. Jin, W. L. Woo, J. R. Woodward, and D. Davies, “Noise tolerant drift detection method for data stream mining,” *Inf. Sci.*, vol. 609, pp. 1318–1333, 2022.

[30] S. C. Hoi, D. Sahoo, J. Lu, and P. Zhao, “Online learning: A comprehensive survey,” *Neurocomputing*, vol. 459, pp. 249–289, 2021.

[31] S. Shalev-Shwartz, “Online learning and online convex optimization,” *Found. Trends Mach. Learn.*, vol. 4, no. 2, pp. 107–194, 2012.

[32] S.-J. Huang, R. Jin, and Z.-H. Zhou, “Active learning by querying informative and representative examples,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 1936–1949, Oct. 2014.

[33] S.-J. Huang, M. Xu, M.-K. Xie, M. Sugiyama, G. Niu, and S. Chen, “Active feature acquisition with supervised matrix completion,” in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 1571–1579.

[34] X. Xu, I. W. Tsang, and D. Xu, “Soft margin multiple kernel learning,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 5, pp. 749–761, May 2013.

[35] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[36] C. Zhu, Z. Wang, and D. Gao, “New design goal of a classifier: Global and local structural risk minimization,” *Knowl.-Based Syst.*, vol. 100, pp. 25–49, 2016.

[37] P.-W. Wang, C.-P. Lee, and C.-J. Lin, “The common-directions method for regularized empirical risk minimization,” *J. Mach. Learn. Res.*, vol. 20, pp. 1–49, 2019.

[38] G. Ausset, S. Clemencon, and F. Portier, “Empirical risk minimization under random censorship,” *J. Mach. Learn. Res.*, vol. 23, pp. 1–59, 2022.

[39] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “Mixup: Beyond empirical risk minimization,” in *Proc. 6th Int. Conf. Learn. Representations*, 2018, pp. 1–13.

[40] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, “MOA: Massive online analysis,” *J. Mach. Learn. Res.*, vol. 11, pp. 1601–1604, 2010.

[41] D. Brzezinski, L. L. Minku, T. Pewinski, J. Stefanowski, and A. Szumaczuk, “The impact of data difficulty factors on classification of imbalanced and concept drifting data streams,” *Knowl. Inf. Syst.*, vol. 63, no. 6, pp. 1429–1469, 2021.

Dianlong You (Member, IEEE) received the PhD degree in computer application technology from Yanshan University, Qinhuangdao, HeBei, China, in 2014. He is an associate professor from 2017 to 2018, he was a visiting scholar with the School of Computing and Informatics, University of Louisiana at Lafayette, Lafayette, Louisiana. His current research interests include machine learning, streaming feature selection and causal discovery.

Jiawei Xiao is currently working toward the MS degree with the School of Information Science and Engineering, Yanshan University, Qinhuangdao, HeBei. His current research interests are focused on streaming feature selection and causal discovery.

Yang Wang is currently working toward the MS degree with the School of Information Science and Engineering, Yanshan University, Qinhuangdao, HeBei. Her current research interests include streaming feature selection and causal discovery.

Huigui Yan is currently working toward the PhD degree with the School of Information Science and Engineering, Yanshan University, Qinhuangdao, HeBei. His current research interests include streaming feature selection and causal discovery.

Di Wu (Member, IEEE) received the PhD degree from the Chongqing Institute of Green and Intelligent Technology, Chinese Academy of Sciences, China, in 2019. He is currently a professor of the College of Computer and Information Science, Southwest University, Chongqing, China. He has more than 60 publications, including journals of *IEEE Transactions on Neural Networks and Learning Systems*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Systems, Man, and Cybernetics*, *IEEE Transactions on Services Computing*, etc., and conferences of ICDM, WWW, IJCAI, etc. His research interests include machine learning and data mining. For more information, please visit https://wudi1989.github.io/Homepage/.

Zhen Chen received the BS and PhD degrees in computer science and technology from Yanshan University, in China, in 2010 and 2017, respectively. He is an associate professor. He is currently working on service computing and data mining.

Limin Shen (Member, IEEE) received the BS and PhD degrees in computer science and technology from Yanshan University, China. He is a professor and PhD supervisor with the College of Computer Science and Engineering, Yanshan University, China. His main research interests include service computing, collaborative computing, and cooperative defense.

Xindong Wu (Fellow, IEEE) received the bachelor's and master's degrees in computer science from the Hefei University of Technology, China, and the PhD degree in artificial intelligence from the University of Edinburgh, Britain, in 1993. He currently is the director and professor of the Key Laboratory of Knowledge Engineering with Big Data (the Ministry of Education of China), Hefei University of Technology, China. He is a Foreign member of the Russian Academy of Engineering, and a fellow of AAAS. He is the Steering Committee chair of ICDM and the editor in-chief of *Knowledge and Information Systems*. His research interests include big data analytics, data mining and knowledge engineering.